WHAT IS CLAIMED IS:

1	1.	A data structure comprising:
2	a 1	nread control block, wherein said thread control block is described by a first
3		data structure; and
4	a ı	nessage, wherein said message is described by a second data structure and
5		said first data structure comprises said second data structure.
1	2.	The data structure of claim 1, wherein said first data structure is
Q.	configure	to store information used to control execution of a thread.
1	3.	The data structure of claim 1, wherein said second data structure is
2	configure	to store a message.
1	4	
1	4.	The data structure of claim 1, wherein said first data structure further
2	comprises	\
3	a j	rocess control block pointer, wherein said process control block pointer
4		points to a process control block;
5	-	cessor information; and
6	sta	ck information.
,		The data structure of claim 4 and are as it are as a control bloods
1	5.	The data structure of claim 4, wherein said process control block
2	comprises	
3		mory information;
4		ead information;
5		vice driver information; and
6	sta	ck information.
1	6.	The data atmestices of alaim 4 subarain said processor information
2	comprises	The data structure of claim 4, wherein said processor information
	•	\
3	_	rocessor identifier; and
4	th	ead information;.

1	7. The data structure of claim 1, wherein said second data structure
2	further comprises:
cub^{3}	control information.
10^{10} 1	8. The data structure of claim 7, wherein said second data structure
2	further comprises:
3	data.
1	9. An operating system, wherein said operating system is configured to
2	provide a user space and a kernel space, comprising:
3 ,	a plurality of tasks, wherein said tasks are executed in said user space;
4	a thread control block/message structure, wherein said thread control
5	block/message structure comprises
6	a thread control block, wherein said thread control block is described
7	by a first data structure, and
. 8	a message, wherein said message is described by a second data
9	structure and said first data structure comprises said second
10	data structure; and
11	a microkernel wherein
12	said microkernel is executed in said kernel space, and
13	said microkernel is configured to support inter-task communication by
14	virtue of being configured to pass said thread control
15	block/message structure from a first one of said tasks to a
16	second one of said tasks
1	10. The operating system of claim 9, wherein said first data structure is
2	configured to store information used to control execution of a thread.
1	11. The operating system of claim 9, wherein said second data structure is

configured to store a message.

2

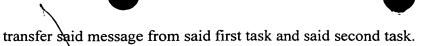
1	12. The operating system of claim 9, wherein said first data structure
2	further comprises:
3	a process control block pointer, wherein said process control block pointer
4	points to a process control block;
5	processor information; and
6	stack information.
1	
2	13. The operating system of claim 12, wherein said process control block
3	comprises:
4	memory information;
5	thread information;
6	device driver information; and
7	stack information.
1	14. The operating system of claim 12, wherein said processor information
2	comprises:
3	a processor identifier; and
4	thread information;.
1	15. The operating system of claim 9, wherein said second data structure
2	further comprises:
3	control information.
1	16. The operating system of claim 15, wherein said second data structure
2	further comprises:
3	data.
1	17. The operating system of claim 9, wherein said operating system
2	employs a client/server architecture.

1	10.	The operating system of claim 17, wherein said first task acts as a
2	client task and	d said second task acts as a server task.
1	19.	A method of inter-task communication comprising:
2	sendir	ng a message between a first task and a second task by performing a send
3		operation, wherein said first task performs said send operation and said
4		send operation employs a thread control block/message structure; and
5	causin	ng said second task to perform a receive operation.
1	20.	The method of claim 19, wherein said thread control block/message
2	structure com	prises:
3	a threa	ad control block, wherein said thread control block is described by a first
4		data structure, and
5	a mess	sage, wherein said message is described by a second data structure and
6		said first data structure comprises said second data structure.
1	22.	The method of claim 19, wherein said thread control block/message
2	structure supp	ports control of a thread within said second task and said method further
3	comprises:	
4	detern	nining if said thread is queued to a thread queue of said second task; and
5	transfe	erring said message from said first task and said second task.
1	23.	The method of claim 21, wherein said transferring said message
2	comprises:	
3		passing said message between said first task and said second task by
4		performing a fast-path message copy it said thread is queued to
5		said thread queue; and
6		passing said message between said first task and said second task by
7		performing a message copy if said thread is not queued to said
8		thread queue.

1	23.	The method of claim 22, wherein said performing said fast-path
2	message copy	comprises.
3		copying said message from a memory space of said first task to a
4		memory space of said second task.
1	24.	The method of claim 22, wherein said performing said message copy
2	comprises:	
3		copying said message from said first task to said thread control
4		block/message structure;
5		waiting for said thread to be queued to said thread queue; and
6		copying said message from said thread control block/message structure
7		to said second task.
1	25.	The method of claim 22, wherein said first task acts as a client task and
2	said second ta	ask acts as a server task.
1	26	A commutes and community and disconnection and delicenses disconnections
1	26.	A computer program product encoded in computer readable media,
2	_	program product comprising:
3	a first	set of instructions, executable on a computer system, configured to send
4		a message between a first task and a second task by performing a send
5		operation, wherein said first task performs said send operation and said
6		send operation employs a thread control block/message structure;
7	a seco	nd set of instructions, executable on said computer system, configured to
8		cause said second task to perform a receive operation.
1	27.	The computer program product of claim 26, wherein said thread
2		message structure comprises:
3	a unea	ad control block, wherein said thread control block is described by a first
4		data structure, and
5	a mess	sage, wherein said message is described by a second data structure and
6		said first data structure comprises said second data structure.

1	28.	The computer program product of claim 26, wherein said thread
2	control block	message structure supports control of a thread within said second task
3	and said com	puter program product further comprises:
4	a third	d set of instructions, executable on said computer system, configured to
5		determine it said thread is queued to a thread queue of said second
6		task; and
7	a four	th set of instructions, executable on said computer system, configured to
8		transfer said message from said first task and said second task.
1	29.	The computer program product of claim 28, wherein said fourth set of
2	instructions c	comprises:
3	a first	subset of instructions, executable on said computer system, configured
4		to pass said message between said first task and said second task by
5		performing a fast-path message copy if said thread is queued to said
6		thread queue; and
7	a seco	ond subset of instructions, executable on said computer system,
8		configured to pass said message between said first task and said second
9 .	. '	task by performing a message copy if said thread is not queued to said
10		thread queue.
1	30.	The computer program product of claim 29, wherein said first subset of
2	instructions c	comprises:
3	a first	sub-subset of instructions, executable on said computer system,
4		configured to copy said message from a memory space of said first task
5		to a memory space of said second task.
1	31.	The computer program product of claim 29, wherein said second
2	subset of inst	ructions comprises:
3	a first	sub-subset of instructions, executable on said computer system,
4		configured to copy said message from said first task to said thread
5		control block/message structure;
		1

6	a second sub-subset of instructions, executable on said computer system,
7	configured to wait for said thread to be queued to said thread queue;
8	and
9	a third sub-subset of instructions, executable on said computer system,
10	configured to copy said message from said thread control
11	block/message structure to said second task.
1	32. The computer program product of claim 29, wherein said first task acts
2	as a client task and said second task acts as a server task.
1	33. A computer system comprising:
2	a processor;
3	computer readable medium coupled to said processor; and
4	computer code, encoded in said computer readable medium, configured to
5	cause said processor to:
6	send a message between a first task and a second task by performing a
7	send operation, wherein said first task performs said send
8	operation and said send operation employs a thread control
9	block/message/structure; and
10	cause said second task to perform a receive operation.
1	34. The computer system of claim 33, wherein said thread control
2	block/message structure comprises:
3	a thread control block, wherein said thread control block is described by a first
4	data structure, and
5	a message, wherein said message is described by a second data structure and
6	said first data structure comprises said second data structure.
1	35. The computer system of claim 33, wherein said thread control
2	block/message structure supports control of a thread within said second task and said
3	computer code is further configured to cause said processor to
4	determine if said thread is queued to a thread queue of said second task; and
	•



36. The computer system of claim 35, wherein said computer code further	er
configured to cause said processor to transfer said message from said first task and	
said second task is further configured to cause said processor to:	
pass said message between said first task and said second task by performing	g a
fast-path message copy if said thread is queued to said thread queue;	;
and	
pass said message between said first task and said second task by performing	g a
message copy if said thread is not queued to said thread queue.	
37. The computer system of claim 36, wherein said computer code further	er
configured to pass said message between said first task and said second task by	
performing a fast-path message copy is further configured to cause said processor to	ɔ :
copy said message from a memory space of said first task to a memory space	е
of said second task	
38. The computer system of claim 36, wherein said computer code further	er
410000	

- configured to pass said message between said first task and said second task by performing a message copy is further configured to cause said processor to:

 copy said message from said first task to said thread control block/message
 - wait for said thread to be queued to said thread queue; and copy said message from said thread control block/message structure to said second task.
- 39. The computer system of claim 36, wherein said first task acts as a client task and said second task acts as a server task.



structure;